I.E.S. Inca Garcilaso

DESARROLLO DE APLICACIONES WEB EN ENTORNO CLIENTE

Ciclo formativo: Desarrollo de Aplicaciones Web **Curso: 2º**

PROYECTO DEL TEMA 10: "Desarrollo de una web funcional completamente en React "

Proyecto: Web React Profesional con Consumo y Envío de Datos a una API

1.- Objetivo general

Desarrollar una aplicación web completa con **React** que simule un producto real, con un diseño moderno, consumo de datos desde una API, navegación multipágina, gestión de estado, eventos y subida de datos mediante formularios.

2.- Descripción del proyecto

Debes crear una web profesional basada en un tema libre, por ejemplo:

- Plataforma de eventos (conciertos, talleres...)
- Catálogo de productos (tienda ficticia)
- App de viajes (destinos, reservas...)
- Gestor de recetas
- Plataforma de empleo
- App de gimnasios/rutinas
- App inmobiliaria
- App de películas/series

El tema es libre, pero debe cumplir todos los requisitos técnicos.

3.- Requisitos obligatorios

a) React + Componentes

- La estructura general puede ser HTML5 (header, nav, main, footer...)
- **Todo contenido dinámico debe estar hecho con componentes React**
- La aplicación debe estar organizada por carpetas (mínimo: components, pages, services, context, styles)

b) Hooks obligatorios

La web debe utilizar correctamente:

- useState. Para controlar estados como:
    - Listas de datos
    - Formularios
    - Carga (loading)
    - Errores
    - Filtros o búsquedas
- useEffect. Para:
    - Consumir datos de una API al cargar páginas
    - Actualizar datos cuando cambie un estado (ej. filtros)
    - Simular comportamientos reales

c) Eventos obligatorios

Se debe usar una variedad de eventos, como mínimo:

- onClick
- onChange
- onSubmit
- onMouseEnter o onMouseLeave (o similar)

Deben tener utilidad real (no decorativos).

d) Consumo de datos desde API (GET)

La web debe consumir datos desde una API REST. Se permite:

- API pública (ej. JSONPlaceholder, TheMovieDB, OpenWeather, etc.)
- API propia creada por el profesor
- API simulada con json-server

Debe haber al menos una página que muestre datos reales de la API en forma de listado.

3.- Elementos de investigación (obligatorios)

a) **Mínimo 4 páginas usando React Router**. La web debe tener al menos **4 páginas diferentes**, accesibles mediante **React Router**, por ejemplo:

- Home
- Listado
- Detalle
- Formulario de creación
- Página de favoritos
- Contacto
- About

Debe existir una navegación clara (Navbar) y rutas definidas.

b) **Uso de useContext para evitar Prop Drilling**. Debe implementarse un **Contexto global** usando:

- createContext
- useContext

Ejemplos válidos:

- Usuario logueado (simulado)
- Tema claro/oscuro
- Favoritos
- Carrito
- Idioma
- Preferencias

No vale usar Context si solo almacena 1 valor irrelevante. Debe tener sentido.

c) **Formulario que suba datos a una API (POST)**. Debe existir al menos **un formulario funcional** que:

- Use useState para controlar los inputs
- Use onSubmit
- Realice una petición **POST** a una API

Puede ser:

- Crear evento
- Añadir receta
- Publicar oferta
- Añadir producto
- Enviar reseña/comentario

4.-Diseño obligatorio (estilo profesional)

La aplicación debe tener:

- Diseño coherente y moderno
- Responsive (mínimo: móvil + escritorio)
- Tipografía y colores consistentes
- Componentes visuales bien alineados
- No se admite diseño "sin estilo"

Se recomienda usar librerías para estilos:

- Tailwind CSS
- Material UI
- Chakra UI
- Bootstrap
- Shadcn/UI
- Styled Components
- ---

5.- Puntos extra (para subir nota)

Se valorará especialmente el uso de librerías externas como:

- Mapas: Leaflet / Mapbox / Google Maps
- Animación: Framer Motion
- Formularios: React Hook Form + Yup
- UI Components: Material UI / Chakra / Shadcn
- Charts: Recharts / Chart.js
- Carruseles: Swiper

- Notificaciones: Toastify

No es obligatorio usar todas ellas

6.- Funcionalidades mínimas sugeridas (para que el proyecto "se sienta real")

Tu web debe incluir al menos 5 de estas:

- Búsqueda
- Filtros por categoría
- Ordenación
- Vista detalle
- Favoritos
- Paginación o "cargar más"
- Mensajes de carga y error
- Confirmación al enviar formulario
- Modal
- Sistema de alertas/toasts

7.- Entrega del proyecto

El proyecto será entregado en una tarea en Moodle destinada para ello. El alumno debe generar un bloc de notas con el enlace de github del mismo. **El proyecto será presentado (todos los alumnos) el miércoles 25 de febrero a partir de las 16:00**. La entrega del mismo se puede realizar hasta una hora antes de la presentación

Se recuerda que el examen de final de esta asignatura se realizará el miércoles 4 de marzo a partir de las 16:00 horas.

Condiciones importantes

No se permite una web "estática" con React pegado encima.

Debe existir una arquitectura real de componentes.

El uso de API y rutas debe ser funcional.

El formulario debe enviar datos realmente (aunque la API sea fake).